

# Simple Kernel Methods for Identification of Cross-Talk and Misrecognition

Jean-Michel Renders<sup>1</sup>, Manny Rayner<sup>2</sup>, Beth Ann Hockey<sup>2</sup>

(1) Xerox Research Center Europe, 6 chemin de Maupertuis, Meylan, 38240, France

Jean-Michel.Renders@xrce.xerox.com

(2) NASA Ames Research Center, Mail Stop T-27A, Moffett Field, CA 94035-1000

mrayner@riacs.edu

## Abstract

Most spoken dialogue systems decide whether to accept or reject results from the speech recognition component by applying a threshold to the associated confidence score. We show how a simple and general method, based on standard approaches to document classification using Support Vector Machines, can give substantially better performance. Experiments carried out on a medium-vocabulary command and control task show a relative reduction in the task-level error rate by about 25% compared to the baseline confidence threshold method.

## 1. Introduction

A standard architecture for a spoken dialogue system typically includes modules that realise the functionalities of speech recognition, semantic analysis, dialogue management and output generation. Speech is converted into words by the speech recogniser, and then into semantic representations by the semantic analyser; the dialogue manager uses this to formulate an abstract response, which is then realised by the output manager.

This picture, however, omits an important component. Since speech input is noisy and speech recognition is uncertain, at least some of the results from the speech recogniser will be unusable, and should ideally be ignored. This is particularly important for systems that use “open mic” (as opposed to “push-to-talk”) speech recognition, where the recogniser is always turned on and listening. In this case, some of the speech from the user (“cross-talk”) will not even be directed to the system, and should certainly not be responded to.

We will refer to the choice between accepting and rejecting output from the speech recogniser as the “accept/reject decision”. Usually, the speech recogniser produces a confidence score as part of its output, and the accept/reject decision is made simply by rejecting utterances whose confidence score is under a specified threshold. A recent example is [1], which reported an accuracy of 9.1% on cross-talk identification using the confidence threshold method.

In this paper, we will show that simple kernel-based approaches, based on standard methods from the document classification literature, can substantially improve on the baseline confidence threshold approach. Experiments were conducted in the framework of the Clarissa system [2, 3], a voice-enabled procedure navigator that has recently been deployed on the International Space Station. Since the whole point of Clarissa is to make it possible for astronaut users to navigate through procedures in an entirely hands- and eyes-free mode, high-quality open mic speech recognition is crucial to the process.

The rest of the paper describes our solution in detail, fo-

cusssing on two key technical problems: choosing an appropriate kernel function, and adjusting the SVM method to take account of the fact that false accepts are generally more harmful than false rejects.

## 2. The Accept/Reject Decision Task

We start by specifying the task more precisely. We can define the following three categories of utterance:

**Type A:** Utterances directed at the system, for which a good interpretation was produced. We will write  $N_A$  for the number of utterances with good interpretations, and  $E_A$  for the number which are rejected.

**Type B:** Utterances directed at the system, and to which the system could in principle respond, but for which the semantic interpretation produced was incorrect. Usually, this is due to faulty recognition. We write  $N_B$  for the number of utterances with bad interpretations, and  $E_B$  for the number which are incorrectly accepted.

**Type C:** Utterances not directed at the system, or directed at the system but to which the system has no adequate way to respond. We write  $N_C$  for the number of cross-talk or out of domain utterances, and  $E_C$  for the number which are incorrectly accepted.

We wish to accept utterances in the first category, and reject utterances in the second and third. If we want to measure performance on the accept/reject task, the most straightforward approach is a simple classification error rate. Ultimately, however, what we are most interested in is measuring performance on the top-level speech understanding task, which includes both the recognition/semantic interpretation task and the accept/reject task described here.

Constructing a sensible metric for the top-level task involves taking account of the fact that some errors are intuitively more serious than others. In a Type A error, the user can most often correct by simply repeating himself. In a Type B or C error, the user will typically have to wait for the system response, realise that it is inappropriate, and then undo or correct it, a significantly longer operation. This analysis suggests that errors should not all be counted equally, but rather be weighted to produce a utility loss function. If we call the weights for the three types  $w_A$ ,  $w_B$  and  $w_C$ , then the weighted utility loss will be

$$w_A E_A + w_B E_B + w_C E_C$$

Assuming that  $w_B \geq w_A$ , we can normalise this by dividing by the maximum possible loss value, giving the normalised loss

function

$$(w_A E_A + w_B E_B + w_C E_C) / (w_B (N_A + N_B) + w_C N_C)$$

It is not easy to give clear justifications for particular choices of weights. One reasonable candidate is  $w_A = 1$ ,  $w_B = 2$  and  $w_C = 3$ . This reflects the observations that it generally appears to take about twice as long to recover from a false accept as a false reject, and that a false accept of a cross-talk utterance affects not only the user, but also the person with whom they are having the side conversation. We will use the “1–2–3” weighting in the rest of this section.

The discrepancy between the (local) loss function associated with SVM-classifier and the top-level function raises the issue of how to align SVM objectives with the top level ones. In this initial work, we simplify the problem by decoupling the speech understanding and accept/reject subtasks, using separate metrics. Since the weighting on the task metric penalises false accepts more heavily than false rejects, we introduce an asymmetric loss function on the SVM score, which weights false accepts twice as heavily as false rejects. We will refer to this as the  $u_2$  metric, and use it as the filtering task metric to compare different parameter settings.

We also need to adjust the definition of the top-level task metric a little, to take account of the fact that some utterances produce no semantic interpretation at all. Obviously, these utterances can only be rejected. We consequently split class B into B0 (no interpretation) and B1 (bad interpretation), and class C into C0 (no interpretation) and C1 (irrelevant or bad interpretation). The weightings for the top-level loss function are then defined by Table 1:

Class	Score	
	Reject	Accept
A	1	0
B0	1	1
B1	1	2
C0	0	0
C1	0	3

Table 1: Definition of the top-level loss function. “Accept” and “Reject” refer to the decision made by the SVM classifier; utterances in the classes B0 and C0 are always rejected, irrespective of the classifier’s decision.

### 3. An SVM Based Approach

There are several information sources which could potentially be used as input to the accept/reject classification problem. So far, we have limited ourselves to the 1-best result returned by the Nuance speech recognition platform [4], which consists of a list of words, each tagged by a numerical confidence value. It seems likely [5] that further improvements could be achieved by using word-lattices; we were however interested to see what could be achieved using a readily available platform. Figure 1 shows examples of typical recognition results, grouped as “positive” (should be accepted), and “negative” (should be rejected).

The usual way to make the accept/reject decision is by using a simple threshold on the average confidence score; the Nuance confidence scores are of course designed for exactly this purpose. Intuitively, however, it should be possible to improve the decision quality by also taking account of the information in the

#### Positive

go:76 to:70 step:55 eleven:34  
repeat:80 the:67 caution:80  
stop:73 talking:65  
no:36  
i:49 meant:73 stop:68 reading:63  
decrease:51 volume:89

#### Negative

stop:4  
five:11  
where:9 are:14 we:37  
list:36 challenge:49 verify:38 it:10 was:65  
mark:17 terse:65 mode:23 for:65 note:20 for:62 ten:20 thousand:41  
column:7 eight:46 is:55 that:83

Figure 1: Examples of recognition results. Each word is tagged by a numerical confidence value. Positive hypotheses should be accepted, negative ones rejected.

recognised words. For instance, the last example under “Negative” in Figure 1 has a fairly high average confidence score, but looks dubious on semantic grounds. Similarly, the fourth example under “Positive” has a low confidence score, but is none the less plausible. We would like to improve our chances of catching examples like these.

The accept/reject decision is clearly a kind of document classification problem. It is well known [6] that margin-based classifiers, especially Support Vector Machines (SVM), get good results for this kind of task. Our original intuition, when beginning this piece of work, was that slightly modified versions of standard SVM-based document-classification methods would also prove appropriate here. There were two key issues we needed to address in order to apply these methods to the new task. First, we had to construct a suitable kernel function, which will define similarity between two recognition results. Second, we had to take account of the asymmetric nature of the cost function.

#### 3.1. Choosing a kernel function

As the output of the speech recognizer is limited to a sequence of words tagged with confidence scores, we have to define some “kernel” between utterances based on this information. The simple bag of word representation, as traditionally used to represent (written) documents, will lose the important confidence information and is unlikely to produce good results (preliminary experiments not reported here showed that this was actually the case). They are three basic degrees of freedom when dealing with such structures:

- Whether or not to take order information into account (this is the distinction between “weighted bag of words” or “weighted word sequence” kernels);
- as there is only one feature per word, how to combine the number of occurrences and the confidence score information (e.g. how to represent “go:76 to:70 step:55 eleven:34 dot:48 eleven:42” )
- taking the “multi-word” (n-grams) information into account.

In our experiments, the first point turns out to be irrelevant in the sense that the order information does not seem to bring enhancements in the metrics used, while consuming much more computation time (one order of magnitude). For the second point, we decided to simply use the “sum” operator for aggregation, even if other operators such as max or min(sum,1) could be used and theoretically motivated. Finally, the third point is solved by using nonlinear polynomial kernels, which automatically explores the space of n-grams (n=degree of the polynomial kernel). In our experiments, degrees higher than 2 did not increase the performance, so that we restricted ourselves to quadratic kernels (and therefore 2-words terms as features).

### 3.2. Making the cost function asymmetric

There are at least two ways to introduce asymmetric costs in standard SVM implementations. In our experiments, we tried both these possibilities.

Recall that SVM is optimizing a mixed criterion which combines classification errors on a training set and a measure of complexity which is related to the margin concept ([7], p. 220). The first approach is to penalize the distance to the margin for misclassified examples more highly for false positives than for false negatives (this is the  $j$  parameter in the svm-light implementation). Note that, with this (standard) simple approach, the algorithm doesn’t really optimize our utility function, but something which is more or less related; this is still a subject of debate in the Machine Learning community (Ref?).

Another way to deal with asymmetric costs is to use calibration techniques: calibration aims at transforming SVM scores into posterior probabilities in a way that is independent from the class priors (basically  $P(s(x) | Class)$  where  $s(x)$  is the score associated with observation  $x$ ). The optimal Bayesian decision can then be adapted, once the new class priors are known ( $P(Class)$ ), as well as error costs. For a binary problem (accept/reject) with equal cost of errors for all negative examples, when the class distribution can be assumed to be the same on both training and test sets, it is sufficient to approximate  $P(Class = A | s(x))$ , as the optimal Bayes decision is then based on minimizing the expected loss function ( $u_2$  in our case): accept the utterance if

$$(2P(Class = B \text{ or } C | s(x))) < P(Class = A | s(x))$$

or, equivalently, if  $P(Class = A | s(x)) > 2/3$ . We used Isotonic Regression to realize the mapping from SVM-scores into (approximate) posterior probabilities.

## 4. Experiments

A corpus of 10409 labelled utterances was used in order to investigate the impact of three factors on classification and task performance:

**Classifier** We used three types of classifier: a simple threshold on the average confidence score; an SVM with a linear kernel; and an SVM with a quadratic kernel. The SVM classifiers used a set of features consisting of the average confidence score together with the weighted bag of words over the total vocabulary.

**Asymmetric Error** We used two different techniques to deal with asymmetric error costs: the  $j$  intrinsic parameter of SVM, and the recalibration procedure using Isotonic Regression. Recall that recalibration aims at optimising the  $u_2$  loss function at SVM classification level, and not

the task-level loss function. Without recalibration, the decision threshold on SVM-scores is 0.

**Recognition** A persistent folklore result, which we also wished to investigate, is that grammar-based language models (GLMs) give better open mic speech understanding performance than conventional statistical language models (SLMs). Clarissa uses a GLM-based recogniser, created using an example-based method driven by a training corpus. Since the same training corpus can be used to construct a normal class N-gram recogniser [8], it was possible to carry out a fair comparison between the two alternate recognition methods. Both recognisers were trained on the same corpus of 3297 utterances, and have vocabularies of 260 words. WER is 6.3% for the GLM version, and 7.4% for the SLM version; semantic error rates are 6.0% for the GLM and 9.6% for the SLM.

For each choice of parameters, we performed 10 random splits (training/test sets) of the initial set of labelled utterances, learned the model on the training sets, and evaluated the loss functions on the corresponding test sets. The final scores were obtained by averaging the loss functions over all 10 runs. Table 2 presents results for the most interesting cases.

## 5. Conclusions

The following conclusions can be drawn from the figures in Table 2. Each of these conclusions was confirmed by hypothesis testing, using the Wilcoxon rank test, at the 5% significance level.

### 5.1. Improvement on baseline performance

The SVM-based method is very considerably better than the baseline method. The average classification error fell from 9.4% for the best baseline configuration (GT-1) to 5.5% for the best SVM-based configuration (GQ-3), a relative improvement of 42%. In particular, the false accept rate for cross-talk and out-of-domain utterances improved from 8.9% (close to the 9.1% cited in [1]) to 4.7%, a 47% relative improvement, while the error rates on the other individual classes also improved. On the task performance metric, the improvement was from 7.0% to 5.4%, or 25% relative.

### 5.2. Kernel types

Quadratic kernels performed better than linear (around 25% relative improvement in classification error); however, this advantage is less marked when considering the task metric (only 3 to 9% relative increase). Though small, the difference is statistically significant. This suggests that meaningful information for filtering lies, at least partially, in the co-occurrences of groups of words, rather than just in isolated words.

### 5.3. Calibration method

In the case of the SLM recogniser, the use of calibration techniques, either by intrinsic SVM-optimisation or by calibration in post-processing do not bring significant improvement. This may be due to the fact that errors on class B utterances are dominant in this model, and calibration is not really able to move the threshold for these “middle”, ambiguous utterances. When using the GLM recogniser, on the other hand, the gain of calibration is now significant: on the  $u_2$  loss function (that calibration aims at minimizing), we attain a 9% relative improvement when using external calibration and 6% when using intrinsic SVM

ID	Rec	Classifier	$j$	Error rates					
				Accept/reject classification					Task
				Classes			All	$u_2$	
				A	B	C			
ST-1	SLM	Threshold	1.0	5.5%	59.1%	16.5%	11.8%	15.1%	10.1%
SL-1	SLM	Linear	1.0	2.8%	37.1%	9.0%	6.6%	8.6%	7.4%
SL-2	SLM	Linear	0.5	4.9%	30.1%	6.8%	7.0%	8.1%	7.2%
SQ-1	SLM	Quad	1.0	2.6%	23.6%	8.5%	5.5%	7.0%	6.9%
SQ-2	SLM	Quad	0.5	4.1%	18.7%	7.6%	6.0%	7.0%	7.0%
SQ-3	SLM	Quad/r	1.0	4.7%	18.7%	6.6%	6.1%	6.8%	6.9%
GT-1	GLM	Threshold	0.5	7.1%	48.7%	8.9%	9.4%	10.7%	7.0%
GL-1	GLM	Linear	1.0	2.8%	48.5%	8.7%	6.3%	8.3%	6.2%
GL-2	GLM	Linear	0.5	4.7%	43.4%	6.0%	6.7%	7.9%	6.0%
GQ-1	GLM	Quad	1.0	2.7%	37.9%	6.8%	5.3%	6.7%	5.7%
GQ-2	GLM	Quad	0.5	4.0%	26.8%	6.0%	5.5%	6.3%	5.6%
GQ-3	GLM	Quad/r	1.0	4.3%	28.1%	4.7%	5.5%	6.1%	5.4%

Table 2: Performance on accept/reject classification and the top-level task, on 12 different configurations of the system. “Threshold” = simple threshold on average confidence; “Linear” = SVM classifier with linear kernel; “Quad” = SVM classifier with quadratic kernel; “Quad/r” = recalibrated version of SVM classifier with quadratic kernel; “A” = in-domain and correct semantic interpretation; “B” = in-domain and incorrect or no semantic interpretation; “C” = out-of-domain; “All” = standard classifier error rate over all data; “ $u_2$ ” = weighted average of classifier error using  $u_2$  weights; “Task” = normalised task metric score; “ $j$ ” = value of svm-light  $J$  parameter.

calibration; on the task metric, gains are reduced to 5% (relative) for external calibration, and only 2% for SVM-calibration (still statistically significant). Error rates on individual classes show that this improvement is due to an increased rate of rejection of class A utterances and a smaller rate of acceptance of both class B and class C utterances.

#### 5.4. Recognition methods

Using the baseline method, there was a large difference in performance between the GLM-based GT-1 and the SLM-based ST-1. In particular, the false accept rate for cross-talk and out-of-domain utterances nearly twice as high (16.5% versus 8.9%) for the SLM-based recogniser. This supports the folklore result that GLM-based recognisers give better performance on the accept/reject task.

When using the SVM-based methods, the best GLM-based configuration (GQ-3) performs about as well as the best SLM-based configuration (SQ-1) in terms of average classification error, with both systems scoring about 5.5%. GQ-3 does perform considerably better than SQ-1 in terms of task error (5.4% versus 6.9%, or 21% relative), but this is due to better performance on the speech recognition and semantic interpretation tasks. Our conclusion here is that GLM-based recognisers do not necessarily offer superior performance to SLM-based ones on accept/reject performance, if a more sophisticated method than a simple confidence threshold is used.

#### 5.5. Relative error rates on individual classes

A deeper look at the error rates by individual classes show that, for all methods, by far the largest error rates are observed for the B category. This is not surprising: experience and intuition tell us that the hardest decision problem is distinguishing between correctly recognised in-domain and incorrectly recognised in-domain utterances. What makes it difficult is that “incorrect” is often better characterised as “partially correct”, so we have much less information to go on.

For example, a problem in any domain which involves

English numbers is the well-know “teen/-ty” ambiguity — “thirty” sounds a lot like “thirteen”, and so on. If the user says “go to step thirty” but the system recognises “go to step thirteen”, the correct action is to reject, but there is little information available to support the decision: basically, we need confidence to be low on the word “thirty”, but our observation is that this is atypical. There are a number of similar generic cases where partial recognition happens easily. We can contrast this with distinguishing between A and C class utterances — cross-talk recognition results tend to look very different from correct in-domain ones, since cross-talk recognition is usually completely wrong rather than just partially wrong. The decision problem is thus intuitively simpler.

## 6. Acknowledgements

Thank you XRCE, NASA etc.

## 7. References

- [1] J. Dowding and J. Hieronymus, “A spoken dialogue interface to a geologist’s field assistant,” in *Proceedings of HLT-NAACL*, Edmonton, Alberta, 2003.
- [2] Clarissa, <http://www.ic.arc.nasa.gov/projects/clarissa/>, 2005, as of 15 February 2005.
- [3] M. Rayner and B. Hockey, “Side effect free dialogue management in a voice enabled procedure browser,” in *Proceedings of INTERSPEECH 2004*, Jeju Island, Korea, 2004.
- [4] Nuance, <http://www.nuance.com>, 2003, as of 25 February 2003.
- [5] C. Cortes, P. Haffner, and M. Mohri, “Lattice kernels for spoken-dialog classification,” in *Proceedings of ICASSP 2003*, Hong Kong, 2003.
- [6] T. Joachims, “Text categorization with support vector machines: Learning with many relevant features,” in *Proceedings of the 10th European Conference on Machine Learning*, Chemnitz, Germany, 1998.

- [7] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
- [8] M. Rayner, P. Bouillon, B. Hockey, N. Chatzichrisafis, and M. Starlander, "Comparing rule-based and statistical approaches to speech understanding in a limited domain speech translation system," in *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*; also [ftp://issco-ftp.unige.ch/pub/publications/tmi\\_045.pdf](ftp://issco-ftp.unige.ch/pub/publications/tmi_045.pdf), Baltimore, MD, 2004.